# Archive Format Requirements for Long Term Storage of Dynamic Signal Data

**A. Phillips, R. Allemang**
University of Cincinnati, College of Engineering and Applied Science, School of Dynamic Systems
PO Box 210072, Cincinnati, OH 45221-0072 USA
email: **ufr@sdrl.uc.edu**

## Abstract

There is a need within the vibration technical community for a long-term viable, open definition file format for the archiving of dynamic signal data and results. This flexible archive format, independent of any particular hardware or operating system environment and distinct from any particular database management structure, is needed in order to satisfy the increasing legal requirements of long term record keeping. For many years, the Universal File Format has been the de facto standard in this area. However, as technology has progressed, the aging nature of this 80 character line, FORTRAN card image based format has become problematic. Following a brief discussion of some of the strengths and weaknesses of existing data formats, this paper focuses upon the identified feature set needed for realistic, long-term reliable recovery of information and successful future community adoption.

## 1 Introduction

It should be noted that, for the purposes of discussion, the existing Universal File Format (UFF) has been taken as the initial starting reference point and this paper is a presentation of a work-in-progress[1]. Various other formats were reviewed and considered for content and applicability; however, in order to facilitate technical community adoption, the final resulting format has been specified to be open, extensible, and non-proprietary. In addition, the principle of 'keeping it simple' has been followed in order to facilitate both industry acceptance and long term comprehension. The recognition is that if it gets too complicated, no one will use, support, or adopt it. For this reason, the final resulting format will probably not be perfect for everyone but it should sufficient for everyone's needs, in other words, a 95% solution. This decision is consistent with the consensus of opinion expressed at a meeting of users and vendors held at IMAC in 1998. The focus of that meeting was to determine interest level and ideas for extending the UFF to address some of its basic deficiencies. In many respects, this project has benefited from and is somewhat of an outgrowth of that activity.

DISCLAIMER

This work of authorship and those incorporated herein were prepared by Contractor as accounts of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor Contractor, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, use made, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency or Contractor thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency or Contractor thereof.

# 2 Guiding Principals

Before discussing the new format, it is important to avoid initial misconceptions by discussing briefly what the new format is not. The new format is focused upon long term archival of dynamic data; as such, things like the data storage media (hardware) and the vendor specific internal database structures are not being addressed. There is no intention or desire to force any particular hardware or internal database structure upon individual vendors or users. The only goal is to produce a long term, viable, cross platform, open architecture, dynamic data storage format. In practical terms, this format should allow users to export data into a data archival structure that is independent of computer operating system and/or the original application program that generated the data and ultimately retrieve the data into other operating systems and other application programs at some later date (up to 50-75 years later if necessary). The realistic need to move the data archive from one media to another over this extended period of time is not a concern of this effort.

It is also important to recognize that the since the content focus is dynamic data, other data content types, such as CAD/CAE, video, pictures, etc., will not be specifically included in the format. It should be noted, however, that although such data will not be specifically identified and targeted for support, nothing in the definition will prevent referencing such information via the meta data records or including it within the data archive container.

One overarching principal however is recoverability! As a long term archive format, any feature or suggestion which jeopardizes recoverability must be subservient. One example of this is the decision to abandon strict backward compatibility with the existing UFF definition; instead, to handle UFF as other data formats via an importer/convertor.

# 3 Purpose

The expected format is intended to fundamentally extend and/or replace the UFF; hence it might be thought of as roughly 'UFF2ish', a sort of second generation UFF. The advantage is that the UFF has been relatively stable and effective for around 40 years. While nobody particularly likes it, nonetheless, as a least common denominator, it has in the past basically gotten the job done. What is needed is to address the core UFF weaknesses that have developed over the years as technology has advanced.

The UFF was developed in the late 1960’s by the Structural Dynamics Research Corporation (SDRC). The original intent was as a cross platform interchange format. It functioned well in that capacity and because of its success it became the default archive format.

One of the known UFF weaknesses is in the area of meta data where there is no mechanism for users to attach arbitrary, test relevant condition information or other pertinent comments to UFF data records in a portable manner. The desired format must include a naturally extensible meta data capability by providing mechanisms for easy, natural extension as new needs develop, while providing backward compatibility, as much as practical. Another known weakness is the aging, eighty ASCII character, FORTRAN card image format. Still another weakness is the serial stream dependency in the UFF definition. (This is most serious in the area of units handling, where a loss or error can cause all succeeding records to be misinterpreted.)

Again, another important point of clarity should be noted: the purpose of the format is primarily archival, not an active database. As a result, the focus of the definition is upon an archive (streamed) format, NOT upon any particular programming language implementation or representation. Retrieval performance of the data is also NOT a primary concern. (Although as computers get faster, discs get bigger, and memory gets cheaper, the issue of adequate performance should be moot.)

The long term goal is to encourage adoption by the dynamics community (both vendor and user) as an export and import format by having a set of libraries in both source and executable format on the University of Cincinnati - Structural Dynamics Research Laboratory (UC-SDRL) web site for use by the community. The UC-SDRL web site will provide a clearing house for enhancements and bug fixes which can be submitted back to UC-SDRL for incorporation into the reference implementations. Currently, the UC-SDRL web site provides documentation for the existing UFF data structures.

Finally, it is the intention that long term there will be a set of software test suites to facilitate compliance and validation checking of implementations. There is no intention to require the community (and vendors in particular) to use the reference implementations in order to achieve compliance. Anyone may develop an optimized version from the specification and validate against the compliance test suite.

# 4 Historical Abuses of the UFF

Over the years, because of misunderstanding of the format definition and because of the uncertainty about handling various data types, there has arisen several frequent and yet understandable abuses of the UFF which cause the files to be less portable than they might otherwise be and effectively non-transportable between different hardware and software systems or even unreadable and unrecoverable. Some of the most notable problems have been:

- Storing critical, non-documentary information in textual ID lines
- Exceeding 80 character line lengths
- Inconsistent, order dependent units issues
- Misunderstanding the format definition
- Invalid field data values and formats (C vs. FORTRAN)
- White space errors (spaces vs. tabs)
- No clear procedure for format error handling
- Lack of user definable fields resulting in storing critical, non-documentary information in textual fields

In all of these examples, the result was a format that became less portable (or even non-transportable) and potentially unreadable or recoverable.

# 5 Current Project Summary

Overall, the project is focused upon the long term archival of dynamic measurement and associated meta data. Performance and size of the archive are not the primary objectives; data integrity and recoverability are the prime objectives. The project is limited to the detection of inadvertent data corruption and extraction of remaining valid data. The problems associated with malicious damage are specifically outside the scope of the project. The issue of refreshing the data, as media storage technology changes, will be required but is also not the concern of this project.

The current plan for the primary data container is to use the ECMA-376 'Open Container'[2] (or close equivalent.) It is essentially a restricted format, industry standard, open architecture ZIP file. The contents are envisioned primarily as sets of XML data streams. The strength of this container is that it can hold structurally organized data and retain the structure. It can also contain and store non-format defined (vendor specific) data.

One of the challenges to this project has been that there are very few data formats that are open and usable without restriction. In the course of the project, three primary candidates were identified for consideration as the format basis: HDF[3], ASAM-ODS ATF-XML[4], and a custom developed XML format. Unfortunately, each has a significant weakness.

- The weakness of the HDF format is that it is essentially a binary file system embedded in a file. Long term damaged data recovery will be problematic.
- The weakness of the ASAM-ODS is that the format prefers external binary files for large data. Using direct file references for these parts makes long term data integrity problematic, potentially placing the entire dataset at risk of complete loss.
- The weakness of a straight XML implementation is that the XML standard requires that any conforming parser stop processing upon encountering any error.

Because the only historically successful, long term archival format is the traditional book, there is a focus upon ASCII/textual data type formats.

# 6 Format Development Activity

The process of reviewing the three primary archival format candidates noted above proceeded, in part, by reviewing existing available data format options with a specific focus upon applicable features for incorporation into the resultant archival format. Since most data formats are targeted at either data transport or active database manipulation, some of their design decisions are at odds with the long term archival goal of the project; nonetheless, many of their specific data features are still relevant. Some of the positive and negative aspects of these different formats were considered in light of these specific features and how long term implementation might be affected.

Consideration was also given to the feature characteristics needed for long term read/recovery viability and industry acceptance. In particular, these two elements favor a format that is principally textual (ASCII) encoded data, which is nominally familiar, is simple to implement and can be mapped relatively straightforward to existing proprietary databases.

Of the formats previously reviewed, the HDF format, while providing the potential mapping structure, is essentially a high performance file system embedded in a file. Besides being relatively complicated, corruption of the data container appears to make recovery of the data difficult. The use of a straight XML format has the advantage of being basically verbose textual (ASCII) information, but unfortunately, the XML standard requires that conforming parsers must halt at any parsing error. Additionally, extraction of any data information requires effectively reading the entire file.

Although proprietary data formats exist which are fundamentally ASCII/binary data interleaved, such formats cannot be considered because of their proprietary nature. However, one format specification standard, developed in the area of textual document interchange, appears to have significant application to this project. It is the 'Office Open XML Format, ECMA-376, Second Edition, Dec. 2008.'

The textual document attributes of headers, footers, cross references, body text, etc. share many conceptual features common to the archiving of dynamic data, that is, data headers, meta data, cross channel references, etc. Hence, the packaging of such data can conceptually be considered a type of dynamic document. The Office Open XML Format and in particular the portion referred to as 'Open Packaging Conventions', includes many of the characteristics of the desired archive data definition. While the specification was primarily developed to support textual documents, the actual specification is general and not specific to such documents. Effectively the definition is a random access container holding primarily textual data. By being based upon familiar industry standards (some being de facto definitions), the format has the potential for easier industry acceptance.

The data recovery features of the potential format are not focused upon deliberate malicious data manipulation, but upon inadvertent corruption. Depending upon the type and degree of corruption, through the use of appropriately tagged prefix meta data (effectively providing redundant container information), the valid uncorrupted data could still be extracted from a damaged archive. Thus, potentially all or most of an archive could be reconstructed in the event of container information corruption.

# 7 Archive Feature Elements

During the various formal and informal discussions with users and vendors that occurred during this project, many suggestions for desirable features were offered which, while not immediately applicable to the initial project effort, were worth noting for consideration during future work. Many of the suggestions do not affect the principal data per se, but rather focus on the retention of "historical" meta information and the like. Examples of these suggestions and concerns are:

- It should be possible to write "noisy" or verbose output (ie. redundant info) with "equivalence" constraint testing capability. (eg. writing multiple measurement vectors from measurement matrix and checking measurement characteristics or constraints. [fmin, deltaf, testid, block length, etc.])
- When preserving data it should be possible to write a "noisy" or verbose output with some form of "backtrace" to the original database fields. (eg. perhaps writing <meas vendorSource="floogle[1]">... data ...</meas> where "floogle[1]" may be the original vendor data ID.)
- It might be advantageous to reserve all 'vendorXXX' attribute fields for vendor use.
- It might also be advantageous to reserve all 'userXXX' attribute fields for end-user use.
- In developing the XML data specification, attributes should not provide any data information, but only meta information about the data.
- It should be possible to tag or log any hardware or software that has "touched" the data. (ie. retain the data history path.)
- It should be possible to document vendor specific or proprietary information within the container using human readable ASCII/XML - *NOT* PDF/DOC/etc.
- The 'Open Container' should allow inclusion of other non-format defined information types. (eg. images, sounds, etc.)
- The format should have clearly defined behavior as well as content. (ie. specified error handling in the presence of malformed data.)
- Inline data should be written in decimal: floating point or bytes. Complex data should be specified as successive pairs of real values.

Although additional feedback is expected as the project continues to progress, these types of comments favor the development of an 'XMLized' UFF-like format definition. Additionally, many of these suggestions are inherently supported by the current concept through the synergy of the ECMA 'Open Container' coupled with a predominantly XML data definition. Further, an ‘XMLized’ UFF has the strength of familiarity, thus facilitating community acceptance.

# 8 Feature Elements Driven by Recoverability

Since the overarching principal is long-term recoverability, many of the archive feature characteristics have been governed by that objective. As mentioned before, the only historically successful, long term archival format has been printed matter. Books, papyri, engravings, etc. all yield valuable (and recoverable) information, even when significantly damaged. The following discussion presents the design impact of recoverability upon some of the archive features.

All data shall be written in UTF-8 encoding to facilitate recovery. Because UTF-8 encoding is backward compatible with ASCII, it guarantees that no low order (0x00-0x7F) ASCII characters occur in any multi-byte encoding, thus the data stream is also self synchronizing. This behavior, coupled with additional constraints, such as requiring all UFR format master control fields to be strict ASCII UPPERCASE (eg. DRT, VER, LREF, XREF, etc.) and requiring all record specific informational fields to be ASCII MixedCase. (eg. DataType, TemperatureOffset, Length, etc.), enables more robust data recovery in the event of inadvertent archive corruption.

To facilitate recovery, large data records should be broken into smaller, more manageable pieces (eg. segments of 50-100 kb.) The various pieces shall be associated using connection references and segment variable features (such as Fmin or Tmin) shall be adjusted to be correct for each segment. For example, the format (structural arrangement) of time series (function) data must have ability to be "sliced-n-diced" throughout the data stream as needed for best resilience against data corruption. The series shall be broken into a set of "manageable" pieces, each with individual checksum coding. The encoding must be distinct from the validation code stored in the ZIP container element header. The series pieces can be organized by any of the following from single complete channel record to multiple channels interleaved (with a granularity [blocksize] from complete record down to single point). As becomes evident, the functional information must intrinsically support multidimensional data.

The archive must contain redundant structural (data organizational) information (preserved with each data record element) in an extractable ASCII readable form. The archive must support redundant (duplicate) data records for key informational content. Also as part of the semantic (informational) structure, each data matrix should receive a unique identification (UID/name) thus also helping to support multiple sets of similar information within the archive and allowing more convenient mapping of vendor database structures.

All field definition (content) strings should be trimmed of leading and trailing white-space. This helps address the issue of the user adding white-space for visual and/or readability purposes, but which is not relevant (or influential) to the information being stored. Thus the ability of the software to read and interpret the informational field correctly is not compromised by a user preference or idiosyncrasy.

Other feature concepts which support recoverability include:

- Each ZIP file entry contains a single archive data record.
- Binary data must NOT be mixed (interspersed) with ASCII (textual) data.
- All basic record field names shall be defined using mixed-case English.
- All archive entry names must be archive root relative.
- All external names must be either archive root relative or file system absolute.
- Each EXT (extension) reference, regardless of being internal or external, must consist of the Path, Name, UID, and Type.

Many of these features have been so chosen in order to facilitate the development and utility of recovery codes capable of scanning a damaged archive and then extracting and reconstructing as much as practical of the original information.

# 9 Format Conceptual Example

The following example is presented for conceptual discussion purposes, giving only an impression of the style of data storage. It is not intended to be complete or to represent any particular likely final implementation.

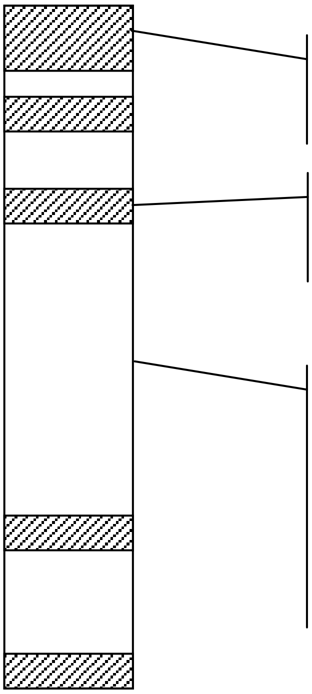
Prototypical Master Field Definition - Information common to all records

| Field Name | Count | Content | Comments |
|---|---|---|---|
| DRT | 1 | String | Data record type or kind |
| VER | 1 | String | Record format revision |
| UID | 1 | String | Unique record identification string |
| SELF | 1 | String | Self-referential archive root relative path |
| EXT | 0-1 | String Array | List of extension record references |

Units Record Example

```
<UFR>
<DRT>Units</DRT><VER>1.0.0</VER><UID>Units-1a</UID>
<SELF>/Measurement/Units/UnitSet1.DSR</SELF><EXT/><System>SI</System>
<Length>1.0</Length><Force>1.0</Force><Temperature>1.0</Temperature>
<TemperatureOffset>273.15</TemperatureOffset>
<TemperatureMode>absolute</TemperatureMode>
</UFR>
```

File Structure Example



container header
[binary information]

container data record header
[binary information]

container data record content

```
<measurement type="FRF" storage="3D" blocklen="512">
<axis type="X" units="Hz" min="0" delta="0.5"></axis>
<axis type="Y" units="g/lbf"></axis>
<DOF type="output" name="NONE" node="1" dir="-Y"/>
<DOF type="input" name="NONE" node="1" dir="Y"/>
<data encoding="base64">badw8...[encoded binary data]...se64Aa5</data>
</measurement>
```

## 10 Conclusions

The design decision of utilizing the ECMA 'Open Container' (or close equivalent) with a predominantly XML data definition is relatively firm as is the decision of developing an XML format that encompasses the familiar UFF characteristics. Still other design decisions yet remain and will be made as the project progresses subject to additional technical feedback; the ultimate goal being to have a relatively final format definition and working reference code completed over the next couple of years.

Please send any comments or suggestions to **UFR@sdrl.uc.edu** .

## References

[1] A. Phillips, R. Allemang, *Requirements for a Long-term Viable, Archive Data Format*, in the *Proceedings of the IMAC-XXVIII Conference & Exposition on Structural Dynamics*, 2010 February 1-4, Jacksonville, Florida USA, 5 pp.

[2] http://www.ecma-international.org

[3] http://www.hdfgroup.org

[4] http://www.asam.net